Project Report

Smart Shop Management System

Guided By :- Anuj Kumar

Name : Aditya Pal
AFID: AF04971732
Batch Code : ANP-D2405
Course Code : ITPR

Name: Pranjal Khokhar
AFID: AF04991772
Batch Code: ANP-D2405
Course Code : ITPR

Table Of Contents

1. Introduction
2. Objectives
3. Project Category
4. System Analysis
    4.1. Modules and Descriptions
    4.2. Database Design
    4.3. Entity-Relationship (ER) Diagram
    4.4. Data Flow Diagram (DFD)
5. Complete System Structure
    5.1. Process Logical Diagram
6. Testing & Results
7. Challenges Faced
8. Platform Used
9. Future Scope
10. Team Work Distribution
11. Conclusion
12. Bibliography

1. Introduction

In the era of digital transformation, retail businesses are rapidly shifting from manual record-keeping to automated systems. Traditional shop management methods suffer from issues such as calculation errors, poor inventory tracking, lack of real-time data, and difficulty in managing taxes like GST. These challenges reduce efficiency and profitability, especially for small and medium-scale retailers.

The **Smart Shop Management System** is a Java-based, console-driven application designed to automate retail operations such as inventory management, customer handling, billing, and sales tracking. The system is developed using **Core Java, JDBC, MySQL, and Maven**, ensuring a professional, scalable, and database-driven architecture. Despite being CLI-based, the application simulates real-world retail workflows accurately and efficiently.

2. Objectives

2.1. Primary Objective

To design and implement a fully functional console-based shop management system that demonstrates real-world software engineering concepts and database integration.

2.2. Specific Objectives

- To automate product, inventory, billing, and customer management.
- To dynamically manage tax configurations (GST slabs).
- To ensure real-time inventory updates after every transaction.
- To implement database normalization and relational integrity.
- To demonstrate practical usage of JDBC, Maven, and SQL.
- To provide a strong academic and industry-oriented project model.

3. Project Category

Application Type: Desktop Console Application (CLI)

Architecture: Layered Architecture (Model, DAO, Utility, Controller)

Database Type: Relational Database Management System (MySQL)

Technology Stack: Java SE, JDBC, MySQL, Maven

4. System Analysis

4.1. Modules and Description

Module 1: Product Management

- Add new products linked with Category, Brand, and Tax
- Update product details (price, brand, tax)
- View product catalog
- Delete obsolete products

Module 2: Inventory Management

- Check available stock
- Refill stock quantities
- Low stock alert system
- Automatic stock deduction after sales

Module 3: Cart & Customer Management

- Customer registration and login
- Product browsing
- Cart operations (add, remove, update quantity)
- Temporary cart handling using Java Collections

Module 4: Billing & Sales Management

- Automatic bill calculation
- Dynamic GST calculation
- Invoice generation
- Sales history storage

4.2. Database Design

The database was implemented using **MySQL 8.0** and consists of **8 normalized tables**. All tables were created programmatically on the first execution of the application using JDBC.

1. **users** – Admin/Staff login credentials

```
mysql> desc users;
+----------+-------------+------+-----+---------+----------------+
| Field    | Type        | Null | Key | Default | Extra          |
+----------+-------------+------+-----+---------+----------------+
| user_id  | int         | NO   | PRI | NULL    | auto_increment |
| username | varchar(50) | NO   | UNI | NULL    |                |
| password | varchar(50) | NO   |     | NULL    |                |
| role     | varchar(20) | NO   |     | NULL    |                |
+----------+-------------+------+-----+---------+----------------+
```

2. **categories** – Product categories

```
mysql> desc categories;
+-------------+--------------+------+-----+---------+----------------+
| Field       | Type         | Null | Key | Default | Extra          |
+-------------+--------------+------+-----+---------+----------------+
| category_id | int          | NO   | PRI | NULL    | auto_increment |
| name        | varchar(50)  | NO   |     | NULL    |                |
| description | varchar(100) | YES  |     | NULL    |                |
+-------------+--------------+------+-----+---------+----------------+
```

3. **brands** – Product brands/manufacturers

```
mysql> desc brands;
+-----------+-------------+------+-----+---------+----------------+
| Field     | Type        | Null | Key | Default | Extra          |
+-----------+-------------+------+-----+---------+----------------+
| brand_id  | int         | NO   | PRI | NULL    | auto_increment |
| brand_name| varchar(50) | NO   |     | NULL    |                |
+-----------+-------------+------+-----+---------+----------------+
```

4. **taxes** – GST and other tax slabs

```
mysql> desc taxes;
+----------+-------------+------+-----+---------+----------------+
| Field    | Type        | Null | Key | Default | Extra          |
+----------+-------------+------+-----+---------+----------------+
| tax_id   | int         | NO   | PRI | NULL    | auto_increment |
| tax_name | varchar(20) | YES  |     | NULL    |                |
| rate     | double      | YES  |     | NULL    |                |
+----------+-------------+------+-----+---------+----------------+
```

5. **products** – Inventory master table

```
mysql> desc products;
+-------------+--------------+------+-----+---------+----------------+
| Field       | Type         | Null | Key | Default | Extra          |
+-------------+--------------+------+-----+---------+----------------+
| product_id  | int          | NO   | PRI | NULL    | auto_increment |
| name        | varchar(100) | NO   |     | NULL    |                |
| price       | double       | NO   |     | NULL    |                |
| quantity    | int          | YES  |     | 0       |                |
| category_id | int          | YES  | MUL | NULL    |                |
| brand_id    | int          | YES  | MUL | NULL    |                |
| tax_id      | int          | YES  | MUL | NULL    |                |
+-------------+--------------+------+-----+---------+----------------+
```

6. **customers** – Customer details

```
mysql> desc customers;
+-------------+--------------+------+-----+---------+----------------+
| Field       | Type         | Null | Key | Default | Extra          |
+-------------+--------------+------+-----+---------+----------------+
| customer_id | int          | NO   | PRI | NULL    | auto_increment |
| name        | varchar(100) | YES  |     | NULL    |                |
| phone       | varchar(15)  | YES  | UNI | NULL    |                |
| email       | varchar(100) | YES  |     | NULL    |                |
| password    | varchar(50)  | NO   |     | NULL    |                |
+-------------+--------------+------+-----+---------+----------------+
```

7. **sales** – Bill header information

```
mysql> desc sales;
+--------------+----------+------+-----+-------------------+-------------------+
| Field        | Type     | Null | Key | Default           | Extra             |
+--------------+----------+------+-----+-------------------+-------------------+
| sale_id      | int      | NO   | PRI | NULL              | auto_increment    |
| sale_date    | datetime | YES  |     | CURRENT_TIMESTAMP | DEFAULT_GENERATED |
| total_amount | double   | YES  |     | NULL              |                   |
| customer_id  | int      | YES  | MUL | NULL              |                   |
+--------------+----------+------+-----+-------------------+-------------------+
```

8. **cart_items** – Bill details

```
mysql> desc cart_items;
+---------------+--------+------+-----+---------+----------------+
| Field         | Type   | Null | Key | Default | Extra          |
+---------------+--------+------+-----+---------+----------------+
| id            | int    | NO   | PRI | NULL    | auto_increment |
| sale_id       | int    | YES  | MUL | NULL    |                |
| product_id    | int    | YES  | MUL | NULL    |                |
| qty           | int    | YES  |     | NULL    |                |
| price_at_sale | double | YES  |     | NULL    |                |
+---------------+--------+------+-----+---------+----------------+
```

Primary and foreign key constraints were enforced to maintain data integrity. Referential relationships were validated through successful CRUD operations during runtime.

4.3 ER Diagram

The ER Diagram represents relationships such as:

Product → Category (N:1) :

- Many Products (e.g., Mouse, Keyboard) belong to One Category (e.g., Electronics).

Product → Brand (N:1)

- Many Products manufactured by One Brand (e.g., Samsung).

Product→ Tax (N:1)

- Many Products falls under One Tax Slab (e.g., GST 18%).

Customer → Sales (1:N)

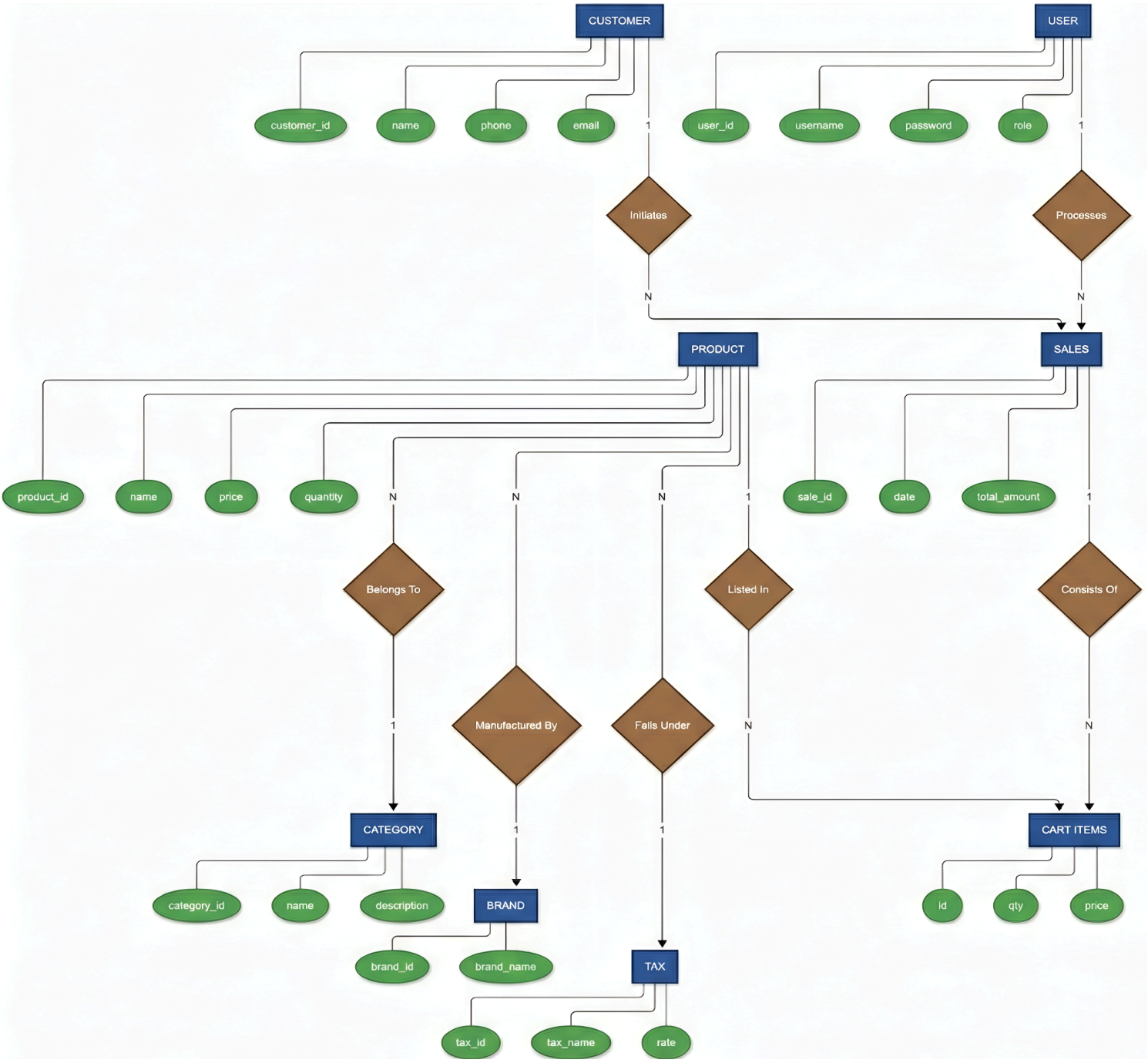
- One Customer *makes* many Sales (Purchases).

Sales → Cart Items (1:N)

- One Sale (Invoice) *contains* many Cart Items (Rows in the bill).

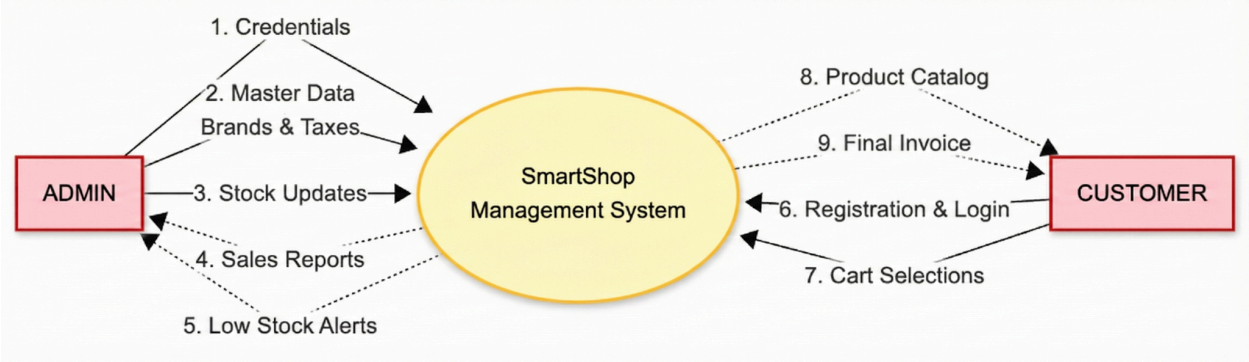
Product → Cart Items (1:N)

- One Product *appears in* many cart Items (Across different bills).

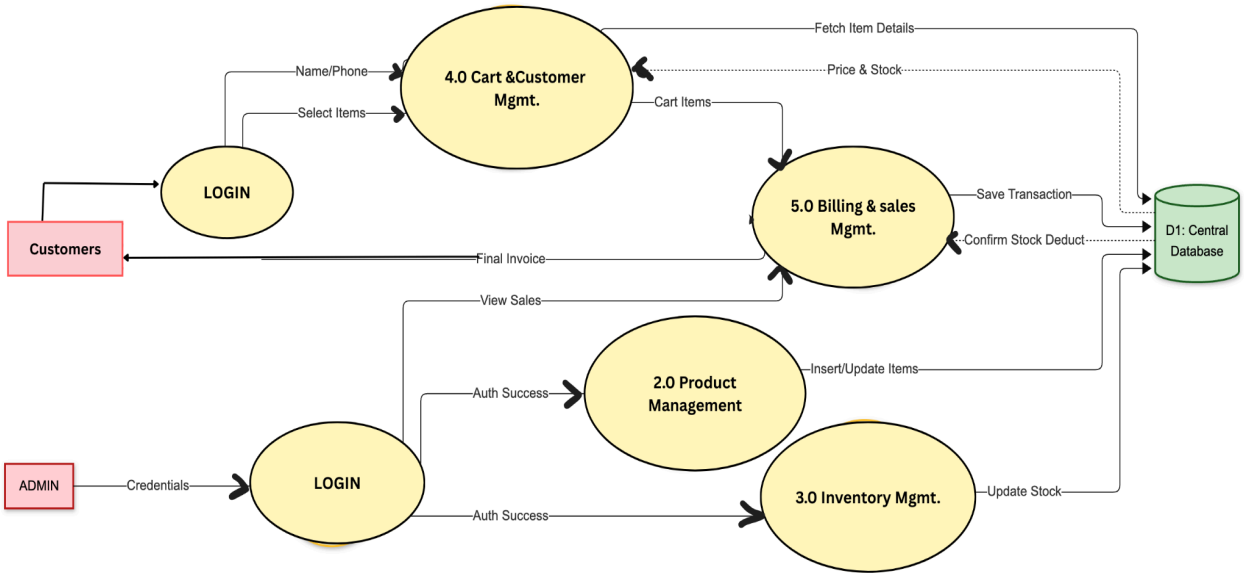

CUSTOMER
USER
customer_id
name
phone
email
1
user_id
username
password
role
1
Initiates
Processes
N
N
PRODUCT
SALES
product_id
name
price
quantity
N
N
N
1
sale_id
date
total_amount
1
Belongs To
Listed In
Consists Of
1
Manufactured By
Falls Under
N
N
CATEGORY
1
1
CART ITEMS
category_id
name
description
BRAND
id
qty
price
brand_id
brand_name
TAX
tax_id
tax_name
rate

4.4 Data Flow Diagram (DFD)

- **Level 0:** Overall system interaction between Admin, Customer, and Database

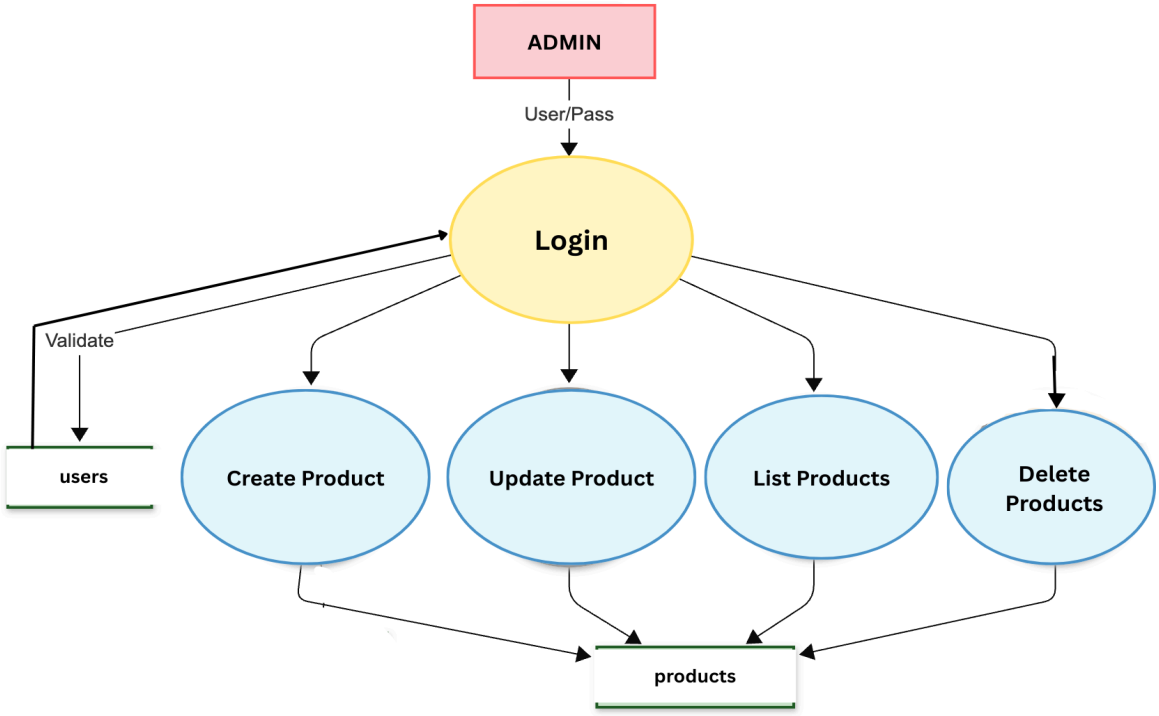


- **Level 1:** Module-wise data flow

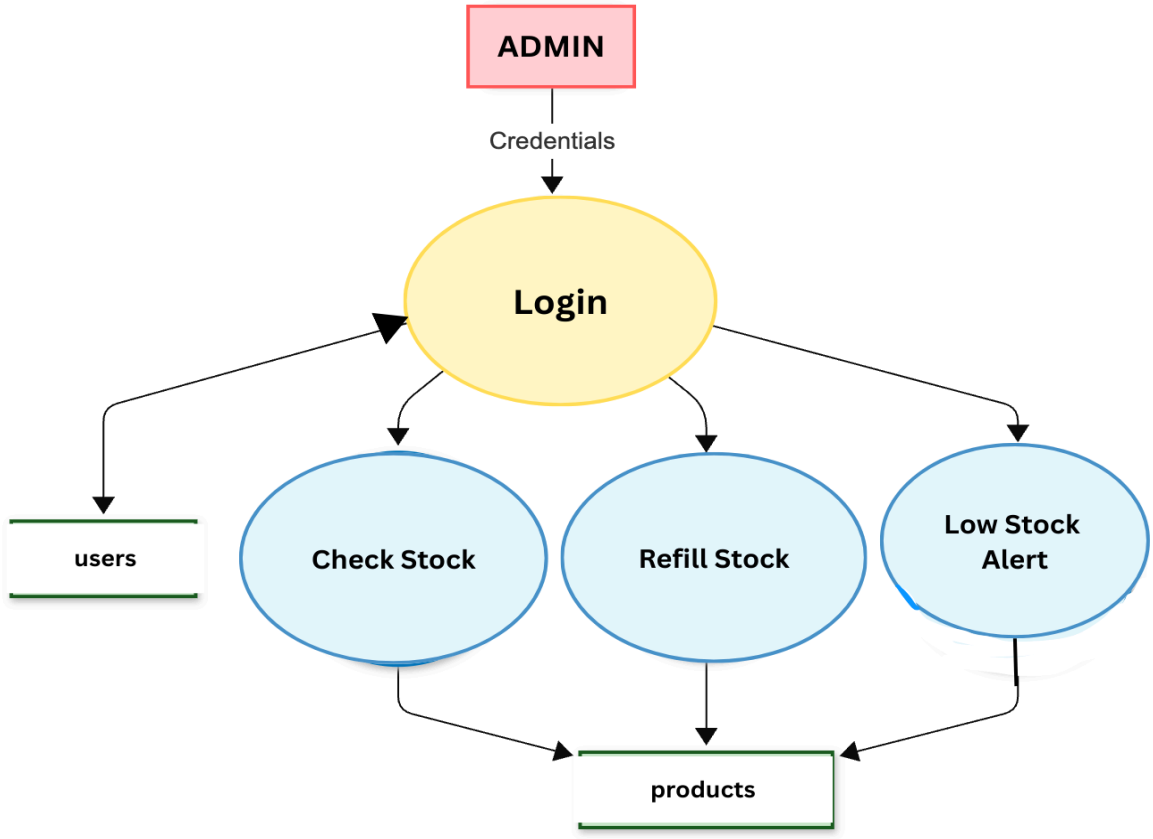


- **Level 2:** Detailed process flow for Product, Inventory, Cart, and Billing modules

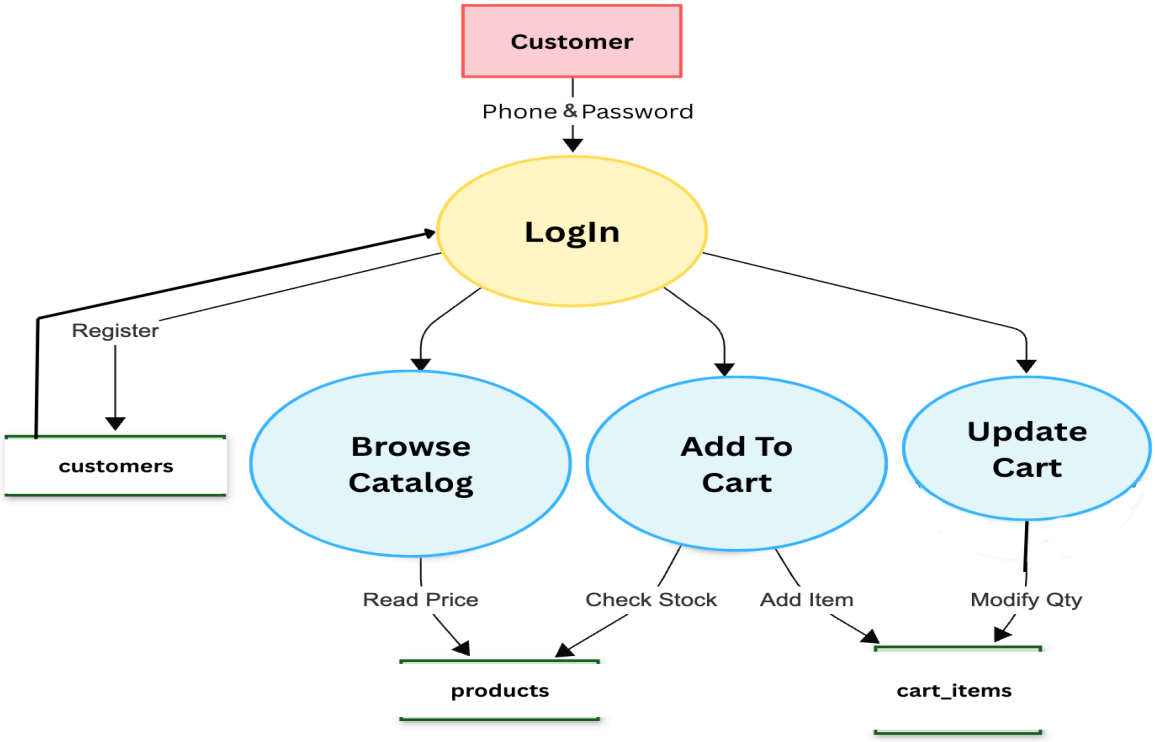
Module 1: Product Management



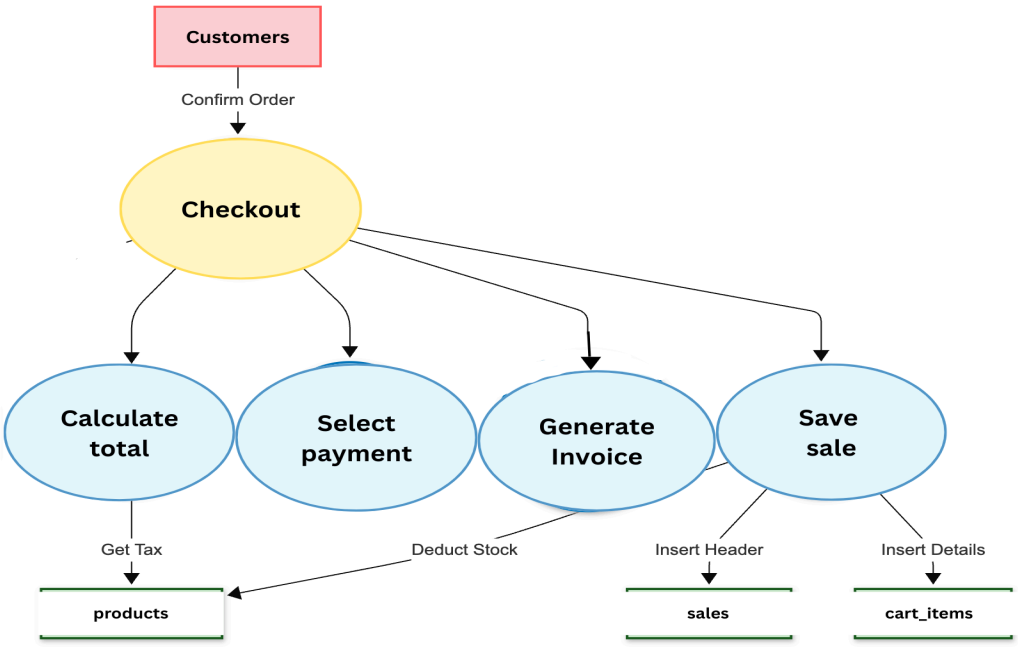
Module 2: Inventory Management



Module 3: Cart & Customer Management



Module 4: Billing & Sales Management



5. Complete System Structure

The project uses a **Layered Architecture** to ensure clean separation of concerns.

5.1 Model Layer

- Represents database tables as Java classes
- Uses encapsulation with getters and setters

5.2 DAO Layer

- Handles all database operations
- Uses JDBC PreparedStatements
- Manages transactions with commit and rollback

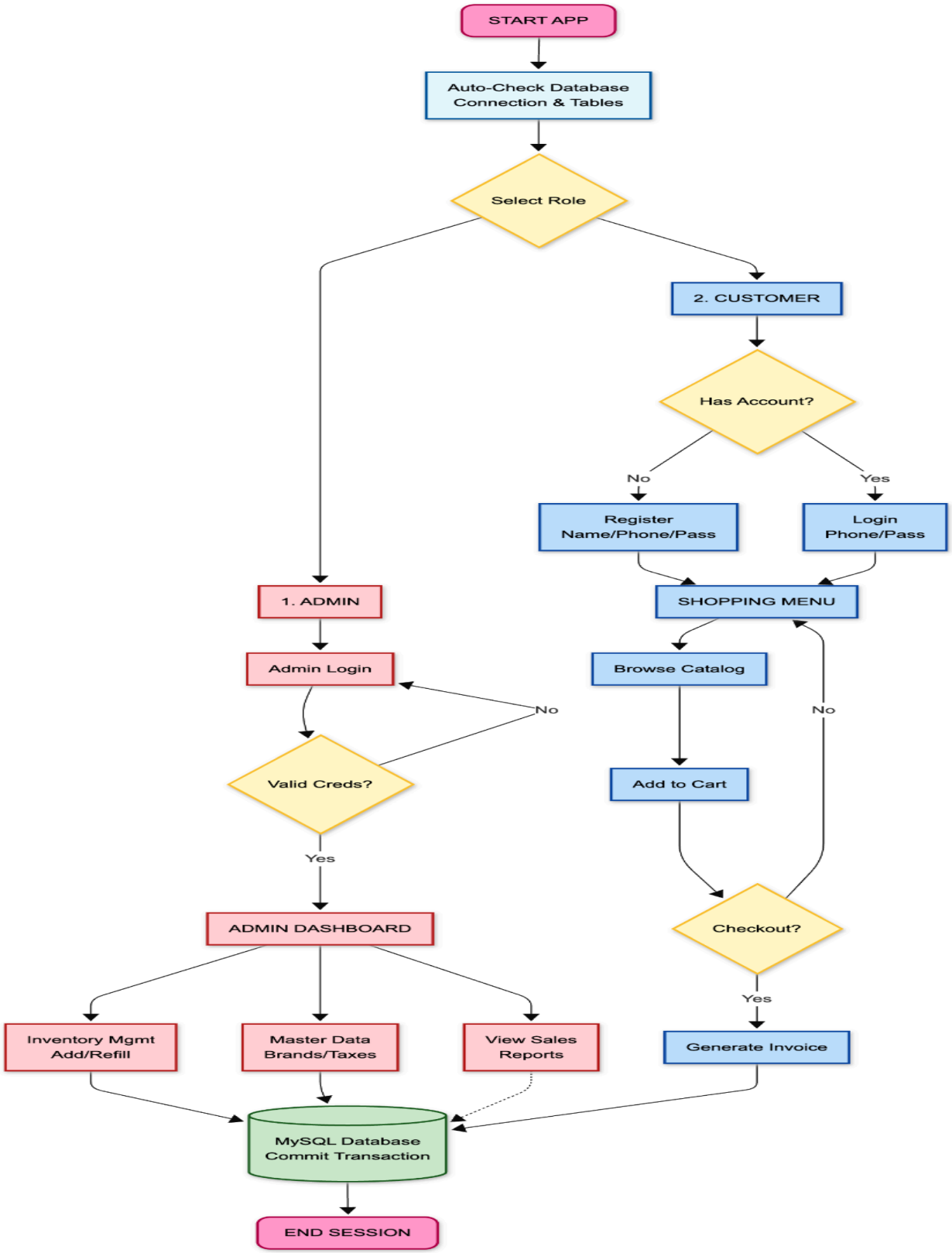
5.3 Utility Layer

- Manages database connectivity
- Automatically creates tables on first run

5.4 Controller Layer

- Application entry point
- Manages user interaction and input validation
- Routes logic between modules

5.1. Process Logical Diagram


START APP
Auto-Check Database Connection & Tables
Select Role
1. ADMIN
2. CUSTOMER
Has Account?
No
Yes
Register Name/Phone/Pass
Login Phone/Pass
SHOPPING MENU
Admin Login
Browse Catalog
No
No
Valid Creds?
Add to Cart
Yes
ADMIN DASHBOARD
Checkout?
Yes
Inventory Mgmt Add/Refill
Master Data Brands/Taxes
View Sales Reports
Generate Invoice
MySQL Database Commit Transaction
END SESSION

6. Testing & Results

- The system was tested using multiple scenarios including:
- Valid and invalid login attempts
- Stock boundary conditions
- Concurrent cart operations
- Successful and failed transactions
- All modules functioned correctly, and the expected outputs were obtained.

7. Challenges Faced

- Managing JDBC transactions across multiple tables
- Maintaining data consistency during billing
- Designing a normalized schema with multiple relationships
- Handling user input validation in a CLI environment

These challenges were resolved through structured design and exception handling.

8. Platform Used

8.1 Hardware Requirements

- Processor: Intel Core i3 or higher
- RAM: Minimum 4 GB
- Storage: 500 MB free space

8.2 Software Requirements

- Operating System: Windows / Linux / macOS
- Java SE (JDK 1.4 or above)
- Apache Maven 3.x
- Eclipse IDE
- MySQL Server 8.0
- MySQL JDBC Connector

9. Future Scope

- Migration to Web-based system using Spring Boot
- Barcode scanner integration
- Online payment gateway support
- Graphical dashboards and reports
- More Role-based access control

10. Team Work Distribution

Aditya Pal

- Database Design
- Product Management Module
- Inventory Management Module
- DAO Implementation

Pranjal Khokhar

- Cart & Customer Management Module
- Billing & Sales Module
- Controller Layer

11. Conclusion

The Smart Shop Management System was successfully designed, implemented, and tested. The project demonstrates practical application of Java, JDBC, and database concepts while solving a real-world retail problem. The system is scalable and can be extended to enterprise-level solutions.

12. Bibliography

- **Core Java:** Concepts of Collections, JDBC, and OOPs.
- **Build Tools:** Apache Maven Documentation (maven.apache.org).
- **Database:** MySQL Normalization principles.
- **MySQL Official Documentation** – https://dev.mysql.com/doc/
- **JDBC MySQL Connector :-** https://dev.mysql.com/downloads/connector/j
- **GeeksforGeeks** – Java & JDBC tutorials
- MySQL Official Documentation (Foreign Keys & Normalization).